Analysis of Branching Algorithms

séminaire AlGco

Serge Gaspers[1]

[1]LIRMM – Université Montpellier 2, CNRS

Nothing is particularly hard if you divide it into small jobs.
– *Henry Ford* (1863–1947)

March 26, 2009

Outline

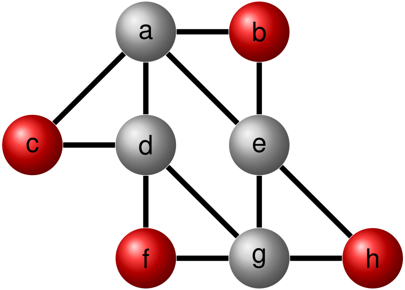
Outline

1. Introduction
2. Simple Analysis
3. Measure Based Analysis
4. Optimizing the measure
5. Search Trees and Branching Numbers
6. Exponential Time Subroutines
7. Towards a tighter analysis
   - Structures that arise rarely
   - State Based Measures
8. Measure Based Analysis for Parameterized Complexity

Maximum Independent Set

Maximum Independent Set (MIS)

- Input: A graph $G = (V, E)$.
- Output: An independent set of G of maximum cardinality.
- $I \subseteq V$ is an independent set if the vertices in I are pairwise non-adjacent.



Branching Algorithm

Branching Algorithm

- Selection: Select a local configuration of the problem instance
- Inspection: Determine the possible values this local configuration can take
- Recursion: Recursively solve subinstances based on these values
- Combination: Compute an optimal solution of the instance based on the optimal solutions of the subinstances

- Reduction: transformation (selection, inspection and the creation of the subinstances for the recursion) of the initial instance into one or more subinstances
- Simplification: reduction to 1 subinstance
- Branching: reduction to ≥ 2 subinstances

Branching Algorithm for MIS

Algorithm **mis**(G)
Input : A graph $G = (V, E)$.
Output: The size of a maximum i.s. of G.

```
if Δ(G) ≤ 2 then                                  // G has max degree ≤ 2
    return the size of a maximum i.s. of G in polynomial time
else if ∃v ∈ V : d(v) = 1 then                    // v has degree 1
    return 1 + mis(G \ N[v])
else if G is not connected then
    Let G_1 be a connected component of G
    return mis(G_1) + mis(G \ V(G_1))
else
    Select v ∈ V s.t. d(v) = Δ(G)                 // v has max degree
    return max (1 + mis(G \ N[v]), mis(G \ v))
```

Outline

Simple Analysis

Lemma 1

Let A be an algorithm for a problem P, and $\alpha > 0,\ c \geq 0$ be constants such that for any input instance I, A reduces I to instances $I_1, \dots, I_k$, solves these recursively, and combines their solutions to solve I, using time at most $\mathcal{O}(|I|^c)$ for the reduction and combination steps (but not the recursive solves) and such that for any reduction done by Algorithm A,

$$(\forall i : 1 \leq i \leq k) \quad |I_i| \leq |I| - 1, \text{ and} \tag{1}$$

$$2^{\alpha\cdot|I_1|} + \cdots + 2^{\alpha\cdot|I_k|} \leq 2^{\alpha\cdot|I|}. \tag{2}$$

Then A solves any instance I in time at most $\mathcal{O}(|I|^{c+1})2^{\alpha\cdot|I|}$.

Simple Analysis for **mis**

- Reduction and combination steps: $\mathcal{O}(n^2)$
- G disconnected:

$$(\forall s : 1 \leq s \leq n-1) \quad 2^{\alpha \cdot s} + 2^{\alpha \cdot (n-s)} \leq 2^{\alpha \cdot n}. \tag{3}$$

  always satisfied by convexity of the function 2^x
- branch on vertex of degree $d \geq 3$

$$(\forall d : 3 \leq d \leq n-1) \quad 2^{\alpha \cdot (n-1)} + 2^{\alpha \cdot (n-1-d)} \leq 2^{\alpha n}. \tag{4}$$

  Dividing all these terms by $2^{\alpha n}$, the constraints become

$$2^{-\alpha} + 2^{\alpha \cdot (-1-d)} \leq 1. \tag{5}$$

Compute optimum α

By standard techniques [Kullmann 99], the minimum α satisfying the constraints is obtained by setting $x := 2^\alpha$, computing the unique positive real root of each of the characteristic polynomials

$$c_d(x) := x^{-1} + x^{-1-d} - 1,$$

and taking the maximum of these roots.

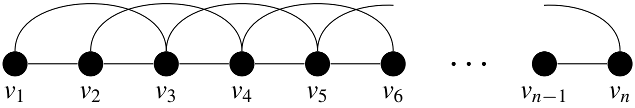
d	x	α
3	1.3803	0.4650
4	1.3248	0.4057
5	1.2852	0.3620
6	1.2555	0.3282
7	1.2321	0.3011

Alternatively, solve a mathematical program minimizing α subject to the constraints (the constraint for $d = 3$ is sufficient as all other constraints are weaker).

Simple Analysis: Result

- use Lemma 1 with $c = 2$ and $\alpha = 0.464959$
- running time of Algorithm **mis** upper bounded by $\mathcal{O}(n^3) \cdot 2^{0.464959 \cdot n} = \mathcal{O}(2^{0.4650 \cdot n})$ or $\mathcal{O}(1.3803^n)$

Lower bound



$$T(n) = T(n-5) + T(n-3)$$

- for this graph, run time is $1.1938\ldots \cdot \text{poly}(n)$
- Run time of algo **mis** is $\Omega(1.1938^n)$

Worst-case running time — a mystery

Mystery

What is the worst-case running time of Algorithm **mis**?

- lower bound $\Omega(1.1938^n)$
- upper bound $\mathcal{O}(1.3803^n)$

Outline

Measure based analysis

- Goal, idea
  - capture more structural changes when reducing an instance to subinstances
- Means
  - potential-function method, such as
    - measure used by [Kullmann 1999],
    - quasiconvex analysis of backtracking algorithms [Eppstein 2004],
    - Measure & Conquer [FominGK 2005],
    - linear programming approach [ScottS 2007], and
    - much older potential-function analyses in mathematics and physics
- Example: Algorithm **mis**
  - advantage when degrees of vertices decrease

Multivariate recurrences

- Model running time of **mis** by

$$T(n_1, n_2, \ldots), \text{ short } T\left(\{n_i\}_{i\geq 1}\right),$$

  where $n_i := |\{v \in V : d(v) = i\}|$.
- $G \setminus v$: neighbors' degree decreases
- $G \setminus N[v]$: a vertex in $N^2[v]$ has its degree decreased

Multivariate recurrences (2)

- We obtain the following recurrence where the maximum ranges over all $d \geq 3$, all $p_i, 2 \leq i \leq d$ such that $\sum_{i=2}^{d} p_i = d$ and all k such that $2 \leq k \leq d$:

$$T\left(\{n_i\}_{i\geq 1}\right) = \max_{d,p_2,\ldots,p_d,k} \begin{cases} T\left(\{n_i - p_i + p_{i+1} - \mathsf{K}_\delta(d=i)\}_{i\geq 1}\right) \\ +T\left(\{n_i - p_i - \mathsf{K}_\delta(d=i) - \mathsf{K}_\delta(k=i) \right. \\ \qquad \left. + \mathsf{K}_\delta(k=i+1)\}_{i\geq 1}\right) \end{cases} \tag{6}$$

where $\mathsf{K}_\delta(F) = \begin{cases} 1 \text{ if } F \text{ true} \\ 0 \text{ otherwise} \end{cases}$

Solve multivariate recurrence

- restrict to max degree 5
- [Eppstein 2004]: there exists a set of weights $w_1, \ldots, w_5 \in \mathbb{R}^+$ such that a solution to (6) is within a polynomial factor of a solution to the corresponding univariate weighted model $(T(\sum_{i=1}^{5} \omega_i n_i) = \max \ldots)$.

Definition 2

A measure μ for a problem P is a function from the set of all instances for P to the set of non negative reals

From recurrences ...

$$\mu(G) := \sum_{i=1}^{5} w_i n_i$$

$$(\forall d : 2 \leq d \leq 5) \quad h_d := \min_{2 \leq i \leq d} \{w_i - w_{i-1}\}$$

By Eppstein, there exist weights w_i such that a solution to (6) corresponds to a solution to the following recurrence, where the maximum ranges over all $d, 3 \leq d \leq 5$, and all $p_i, 2 \leq i \leq d$, such that $\sum_{i=2}^{d} p_i = d$,

$$T(\mu(G)) = \max_{d, p_2, \ldots, p_d, k} \begin{cases} T\left(\mu(G) - w_d - \sum_{i=2}^{d} p_i \cdot (w_i - w_{i-1})\right) \\ +T\left(\mu(G) - w_d - \sum_{i=2}^{d} p_i \cdot w_i - h_d\right). \end{cases}$$

... to constraints

$$T\left(\mu(G)\right) \geq T\left(\mu(G) - w_d - \sum_{i=2}^{d} p_i \cdot (w_i - w_{i-1})\right)$$
$$+ T\left(\mu(G) - w_d - \sum_{i=2}^{d} p_i \cdot w_i - h_d\right)$$

for all $d, 3 \leq d \leq 5$, and all $p_i, 2 \leq i \leq d$, such that $\sum_{i=2}^{d} p_i = d$.

Measure Based Analysis

Lemma 3

Let A be an algorithm for a problem P, $c \geq 0$ be a constant, and $\mu(\cdot), \eta(\cdot)$ be two measures for the instances of P, such that for any input instance I, A reduces I to instances $I_1, \ldots, I_k$, solves these recursively, and combines their solutions to solve I, using time at most $\mathcal{O}(\eta(I)^c)$ for the reduction and combination steps (but not the recursive solves) and such that for any reduction done by Algorithm A,

$$(\forall i) \quad \eta(I_i) \leq \eta(I) - 1, \text{ and} \tag{7}$$

$$2^{\mu(I_1)} + \ldots + 2^{\mu(I_k)} \leq 2^{\mu(I)}. \tag{8}$$

Then A solves any instance I in time at most $\mathcal{O}(\eta(I)^{c+1})2^{\mu(I)}$.

Applying the lemma

$$2^{\mu(G)} \geq 2^{\mu(G)-w_d-\sum_{i=2}^{d} p_i\cdot(w_i-w_{i-1})} + 2^{\mu(G)-w_d-\sum_{i=2}^{d} p_i\cdot w_i-h_d}$$
$$1 \geq 2^{-w_d-\sum_{i=2}^{d} p_i\cdot(w_i-w_{i-1})} + 2^{-w_d-\sum_{i=2}^{d} p_i\cdot w_i-h_d}$$

i	w_i	h_i
1	0	0
2	0.25	0.25
3	0.35	0.10
4	0.38	0.03
5	0.40	0.02

With these values for w_i, the constraints are satisfied and $\mu(G) \leq 2n/5$ for any graph of max degree ≤ 5.
Taking $c = 2$ and $\eta(G) = n$, Lemma 3 shows that **mis** has run time $\mathcal{O}(n^3)2^{2n/5} = \mathcal{O}(1.3196^n)$ on graphs of max degree ≤ 5.

Outline

Compute optimal weights

- random local search [Fomin, Grandoni, Kratsch 2005, 2007]
- quasiconvex programming [Eppstein 2004, 2006]
- convex programming [Gaspers, Sorkin 2009]

All constraints are already convex, except conditions for h_d

$$(\forall d : 2 \leq d \leq 5) \quad h_d := \min_{2 \leq i \leq d} \{w_i - w_{i-1}\}$$

$$\Downarrow$$

$$(\forall i, d : 2 \leq i \leq d \leq 5) \quad h_d \leq w_i - w_{i-1}.$$

Use existing convex programming solvers to find optimum weights.

convex program in AMPL

```
param maxd integer >= 3;
set DEGREES := 0..maxd;
var W {DEGREES} >= 0;  # weight for vertices according to their degrees
var g {DEGREES} >= 0;  # weight for degree reductions from deg i
var h {DEGREES} >= 0;  # weight for degree reductions from deg \le i
var Wmax;              # maximum weight of W[d]

minimize Obj: Wmax;    # minimize the maximum weight

subject to MaxWeight {d in DEGREES}:
  Wmax >= W[d];
subject to gNotation {d in DEGREES : 2 <= d}:
  g[d] <= W[d]-W[d-1];
subject to hNotation {d in DEGREES, i in DEGREES : 2 <= i <= d}:
  h[d] <= W[i]-W[i-1];
subject to Deg3 {p2 in 0..3, p3 in 0..3 : p2+p3=3}:
  2^(-W[3] -p2*g[2] -p3*g[3]) + 2^(-W[3] -p2*W[2] -p3*W[3] -h[3]) <=1;
subject to Deg4 {p2 in 0..4, p3 in 0..4, p4 in 0..4 : p2+p3+p4=4}:
  2^(-W[4] - p2*g[2] - p3*g[3] - p4*g[4])
+ 2^(-W[4] - p2*W[2] - p3*W[3] - p4*W[4] - h[4]) <=1;
subject to Deg5 {p2 in 0..5, p3 in 0..5, p4 in 0..5, p5 in 0..5 :
                 p2+p3+p4+p5=5}:
  2^(-W[5] - p2*g[2] - p3*g[3] - p4*g[4] - p5*g[5])
+ 2^(-W[5] - p2*W[2] - p3*W[3] - p4*W[4] - p5*W[5] - h[5]) <=1;
```

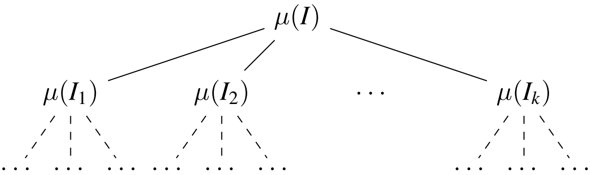
Optimal weights

i	w_i	h_i
1	0	0
2	0.206018	0.206018
3	0.324109	0.118091
4	0.356007	0.031898
5	0.358044	0.002037

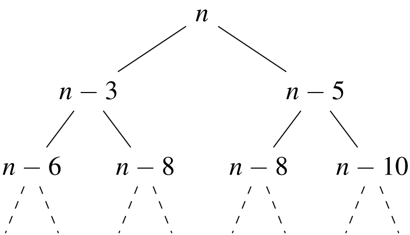
- use Lemma 3 with $\mu(G) = \sum_{i=1}^{5} w_i n_i \leq 0.358044 \cdot n$, $c = 2$ and $\eta(G) = n$
- **mis** has run time $\mathcal{O}(n^3)2^{0.358044 \cdot n} = \mathcal{O}(1.2817^n)$

Outline

Search Trees



Example: execution of **mis** on a P_n^2



Branching number: Definition

Given a constraint

$$2^{\mu(I)-a_1} + \cdots + 2^{\mu(I)-a_k} \leq 2^{\mu(I)},$$

its *branching number* is

$$2^{-a_1} + \cdots + 2^{-a_k},$$

and is denoted by

$$(a_1, \ldots, a_k) .$$

Clearly, any constraint with branching number at most 1 is satisfied.

Branching numbers: Properties

Dominance For any a_i, b_i such that $a_i \geq b_i$ for all i, $1 \leq i \leq k$,

$$(a_1, \ldots, a_k) \leq (b_1, \ldots, b_k),$$

as $2^{-a_1} + \cdots + 2^{-a_k} \leq 2^{-b_1} + \cdots + 2^{-b_k}$.
In particular, for any $a, b > 0$,

$$\text{either} \quad (a, a) \leq (a, b) \quad \text{or} \quad (b, b) \leq (a, b).$$

Balance If $0 < a \leq b$, then for any ε such that $0 \leq \varepsilon \leq a$,

$$(a, b) \leq (a - \varepsilon, b + \varepsilon)$$

by convexity of 2^x.

Outline

1. Introduction
2. Simple Analysis
3. Measure Based Analysis
4. Optimizing the measure
5. Search Trees and Branching Numbers
6. **Exponential Time Subroutines**
7. Towards a tighter analysis
   - Structures that arise rarely
   - State Based Measures
8. Measure Based Analysis for Parameterized Complexity

Exponential time subroutines

Lemma 4

Let A be an algorithm for a problem P, B be an algorithm for (special instances of) P, $c \geq 0$ be a constant, and $\mu(\cdot), \mu'(\cdot), \eta(\cdot)$ be three measures for the instances of P, such that for any input instance I, $\mu'(I) \leq \mu(I)$ and for any input instance I, A either solves P on I by invoking B with running time at most $\mathcal{O}(\eta(I)^{c+1})2^{\mu'(I)}$, or reduces I to instances $I_1, \ldots, I_k$, solves these recursively, and combines their solutions to solve I, using time at most $\mathcal{O}(\eta(I)^c)$ for the reduction and combination steps (but not the recursive solves) and such that for any reduction done by Algorithm A,

$$(\forall i) \quad \eta(I_i) \leq \eta(I) - 1, \text{ and} \tag{9}$$

$$2^{\mu(I_1)} + \cdots + 2^{\mu(I_k)} \leq 2^{\mu(I)}. \tag{10}$$

Then A solves any instance I in time $\mathcal{O}(\eta(I)^{c+1})2^{\mu(I)}$.

Algorithm **mis** on general graphs

- use Lemma 4 with $A = B =$ **mis**, $c = 2$, $\mu(G) = 0.35805n$, $\mu'(G) = \sum_{i=1}^{5} w_i n_i$, and $\eta(G) = n$
- for every instance G, $\mu'(G) \leq \mu(G)$ because $\forall i, w_i \leq 0.35805$
- for each $d \geq 6$,

$$(0.35805, (d+1) \cdot 0.35805) \leq 1$$

- Thus, Algorithm **mis** has running time $\mathcal{O}(1.2817^n)$ for graphs of arbitrary degrees

Outline

Rare Configurations

- Branching on a local configuration C does not influence overall running time if C is selected only a constant number of times on the path from the root to a leaf of any search tree corresponding to the execution of the algorithm
- Can be proved formally by using measure

$$\mu'(I) := \begin{cases} \mu(I) + c & \text{if } C \text{ may be selected in the current subtree} \\ \mu(I) & \text{otherwise.} \end{cases}$$

Avoid branching on regular instances in **mis**

else

Select $v \in V$ such that

(1) v has maximum degree, and

(2) among all vertices satisfying (1), v has a neighbor of minimum degree

return $\max\left(1 + \textbf{mis}(G \setminus N[v]), \textbf{mis}(G \setminus v)\right)$

New measure:

$$\mu'(G) = \mu(G) + \sum_{d=3}^{5} \mathrm{K}_\delta(G \text{ has a } d\text{-regular subgraph}) C_d$$

where $C_d, 3 \leq d \leq 5$, are constants.

Resulting Branching numbers

For each $d, 3 \leq d \leq 5$ and all $p_i, 2 \leq i \leq d$ such that $\sum_{i=2}^{d} p_i = d$ and $p_d \neq d$,

$$\left(w_d + \sum_{i=2}^{d} p_i \cdot (w_i - w_{i-1}), w_d + \sum_{i=2}^{d} p_i \cdot w_i + h_d\right).$$

All these branching numbers are at most 1 with the optimal set of weights on the next slide

Result

i	w_i	h_i
1	0	0
2	0.207137	0.207137
3	0.322203	0.115066
4	0.343587	0.021384
5	0.347974	0.004387

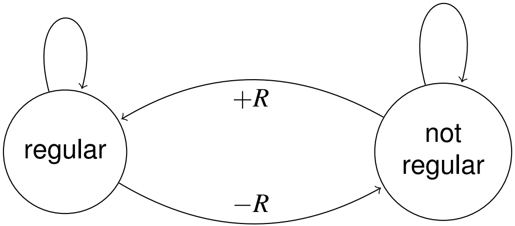
Thus, the modified Algorithm **mis** has running time $\mathcal{O}(2^{0.3480 \cdot n}) = \mathcal{O}(1.2728^n)$.

State based measures

- “bad” branching always followed by “good” branchings
- amortize over branching numbers

$$\mu'(I) := \mu(I) + \Psi(I),$$

where $\Psi : \mathcal{I} \to \mathbb{R}^+$ depends on global properties of the instance.



Outline

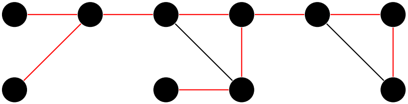
Measure in Parameterized Complexity

- So far: only State Based Measure
- e.g. Wahlström's 3-HITTING SET algorithm analysed with measure $k - \Psi(I)$ where $\Psi : \mathcal{I} \to \mathbb{R}^+$ depends on the number of 2-sets
- Here: use unrestricted measure

MIST

Definition 5

Max Internal Spanning Tree (MIST): Given a graph $G = (V, E)$ and a parameter k, does G have a spanning tree with at least k internal nodes?



We consider MIST on graphs of maximum degree 3.

Preliminaries

Lemma 6 (Prieto, Sloper 2005)

An optimal solution T_o to MIST is a Hamiltonian path or the leaves of T_o are independent.

Lemma 7

MIST on cubic graphs has a $(2k+2)$ kernel.

Hamiltonian Path can be solved in time $\mathcal{O}(1.251^n) = 1.5651^k n^{\mathcal{O}(1)}$ on cubic graphs.

Measure

$$\mu(G,T,k) := k - \omega|X| - |Y|, \text{ where}$$

$X := \{v \in V \mid d_G(v) = 3, d_T(v) = 2\}$,
$Y = \{v \in V \mid d_G(v) = d_T(v) \geq 2\}$, and
$\omega = 0.45346$.

Analyse configurations to obtain branching factors $(\omega, 1)$, $(2-\omega, 1-\omega)$ and $(1-\omega, 2-\omega, 2-\omega)$ (see blackboard).

Theorem 8

MIST can be solved in time $2.7321^k n^{\mathcal{O}(1)}$ on cubic graphs.

Thank you!

Questions?

Comments?